Development in Code_Aster
Creating a command

Code_Aster, Salome-Meca course material
GNU FDL licence (http://www.gnu.org/copyleft/fdl.html)
What is a command?

Different types of command
- Procedures PROC that return no result
  DEBUG, IMPR_RESU
- Operators OPER that return exactly one result
  STAT_NON_LINE, CALC_CHAMP

Each command has its own syntax description
- In catapy/commande/*.capy
- See the presentation of Macro-Command for details

OPER & PROC are written in Fortran
- The main subroutine is op00NN.F90 defined by op=NN in the « capy » file
Passing the keywords values, see d6.03.01

Get the result name returned by the command

```plaintext
call getres( result, type_of_the_result, command )
```

Get the number of occurrences of a factor keyword

```plaintext
call getfac( 'FACT', number_of_occurrences )
```

Get the number of values provided by a simple keyword

```plaintext
call getvr8( 'FACT', 'SIMP', iocc, nbval=0, nbret=size)
```

- `size` will be a negative number (see documentation), the number of values to read is \(-size\)
- Types: getvis (integer), getvr8 (float), getvc8 (complex), getvtx (string), getvid (objects)
- `FACT=' ' for first level keywords

Fill an array with the values of a simple keyword

```plaintext
size = -size
```

```plaintext
call getvr8( 'FACT', 'SIMP', iocc, nbval=size, vect=vector)
```

Or for a scalar:

```plaintext
call getvr8( 'FACT', 'SIMP', iocc, scal=value)
```
Exercise

Create a command, called MODI_MAIL, that translate a mesh by a vector.

- **Inputs:**
  - **MAILLAGE:** The mesh to translate (maillage_sdaster object)
  - **TRANSLATION:** a factor keyword with a unique simple keyword VECTEUR which gives the translation vector provided by 3 float numbers
  - **INFO:** Verbosity flag. Optional, 1 or 2, default is 1.

- **Output**
  - The same mesh, changed in place

- **Optional improvements**
  - Support of 2D and 3D translation vector (2 or 3 values)
  - Use a INFO keyword to print, for example, a message with the number of nodes of the mesh

```
cd $HOME/dev/codeaster/src
hg pull -r 61032c4 codeaster_push
hg update -C 61032c4
```
DEBUT()

mesh = LIRE_MAILLAGE(FORMAT='MED')

mesh = MODI_MAIL(reuse=mesh,
               MAILLAGE=mesh,
               TRANSLATION=_F(VECTEUR= (1., 2., 3.),),)

# check that the coordinates were correctly changed
TEST_RESU(...)

FIN()
Skeleton of modi_mail.capy

```c
MODI_MAIL=OPER(
    nom="MODI_MAIL",
    op=190,
    sd_prod=maillage_sdaster,
    fr=tr("Modifier un maillage par translation"),
    reentrant='o',
    MAILLAGE=TODO,
    TRANSLATION=TODO,
    INFO=SIMP(statut='f', typ='I', defaut=1, into=(1, 2)),
)
```
subroutine op0190()
...
! read the input mesh name: maillage_sdaster, see d6.03.01, §2.1.1
call getvid(,,, nbret=iret)
ASSERT(iret == 1)

! read the mesh result (must be identical to the input), see d6.03.01, §2.1.5
call getres(,...)
ASSERT(mesh == mesh0)

! check that TRANSLATION exists, see d6.03.01, §2.1.6
call getfac('TRANSLATION', nbocc)
ASSERT(nbocc == 1)

! get the size of the translation vector for a dynamic allocation, see d6.03.01, §2.1.1
call getvr8(,..., nbret=dim)
dim = -dim
ASSERT( 2 <= dim .and. dim <= 3)
allocate the vector of size 'dim'
AS_ALLOCATE(...)

read the translation vector values
call getvr8(..., nbval=dim, vect=...)

name of the jeveux vector containing the coordinates of the mesh
see d4.06.01 for the COORDO object and d4.06.05 for its VALE vector
... vect_coord = ...

get the address and the size of this vector
call jeveuo(...)
call jelira(...) nbnode = size / 3

translate the mesh
! loop on the nodes
...